REMARKS

Claims 1-11, 28-33, 41-43 and 47 are currently pending in the subject application and are presently under consideration. Favorable reconsideration of the subject patent application is respectfully requested in view of the comments and amendments herein.

I. Interview Summary Dated August 30, 2006

Applicant's representative thanks the Examiner for extending the courtesy of an interview. The Examiner's Interview Summary indicates that "a limitation containing a language neutral feature will be added to the claim body." (*See* Interview Summary dated August 30, 2006). As discussed in the interview, the phrase "expressed in a language neutral fashion" recited in the preamble of claim 1 prior to the amendments submitted herewith is a statement of purpose and did not serve to the limit the claims. This phrase has been deleted from the preamble as was discussed during the interview. As also discussed during the interview, the claims are patentable over the cited art because the cited art does not disclose expressing an association between a declaration and an implementation in an intermediate language representation and providing this intermediate language representation to a target runtime. However, as also was discussed during the interview, in order to expedite prosecution, the claim term "expressing" has been replaced with the term "compiling" in order to further clarify the claimed subject matter.

II. Rejection of Claims 1-11, 28-34, 41-43 and 47 Under 35 U.S.C §112

Claims 1-11, 28-34, 41-43 and 47 stand rejected under 35 U.S.C. §112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention. Claims 1, 28, 41 and 47 have been amended and accordingly, applicant's representative respectfully requests that this rejection be withdrawn.

III. Rejection of Claims 41-43 Under 35 U.S.C. §101

Claims 41-43 stands rejected under 35 U.S.C. §101. Applicant's representative respectfully submits that these claims are patentable as previously submitted. However, in order to expedite prosecution, these claims have been amended. Accordingly, applicant's representative respectfully requests that this rejection be withdrawn.

IV. Rejection of Claims 1-11, 28-34, 41-43 and 47 Under 35 U.S.C. §102(e)

A. Background

Software programs are commonly written in a high-level programming language, such as VISUAL BASIC, C++, COBOL, Pascal, Smalltalk, or the like. The high-level language statements or instructions of the program (*e.g.*, source code) are then translated or compiled into coded instructions (*e.g.*, native or object code), which are executable by the computer. Typically, a software program known as a compiler is used for this translation. Processes for compiling source code into executable object or native code are well known in the art. The compiler initially performs lexical analysis on the source code to separate the source code into various lexical structures of the programming language (generally known as tokens), such as keywords, identifiers, operator symbols, punctuation, and the like. Syntactical analysis is then performed in which the compiler groups the tokens into various syntax structures of the programming language, such as expressions, declaration statements, loop statements, procedure calls, and the like. Thereafter, the compiler generates and optimizes code for each of these structures.

Such source code program representations may include declarations and implementations which the compiler must associate. For instance, a source program may include a function or method declaration in an interface being implemented by a class. The class implementing the interface may in turn provide an implementation for the declared function or method. The compiler associates the declaration and the implementation in generating native code for the program. Compilers typically employ one or more association rules particular to the source code language of interest in associating such source code declarations and implementations.

Where the programmer writes source code which is vague or ambiguous as to what implementation is to be associated with a particular declaration, the compiler typically generates a compiler error, which the programmer must address before the program can be successfully compiled. In this case, the compiler employs one or more association or disambiguation rules (*e.g.*, signature matching, calling context, etc.), or criteria in determining when to generate such an error. Such association or disambiguation rules are generally specific to a particular source code language, as the syntax and semantics of source

code languages varies greatly. Thus, the association rules appropriate for one source code language may not be (*e.g.*, and typically are not) adaptable to properly associated declarations and implementations in another source code language.

Recent developments in programming technologies have provided common language runtime systems, in which programs and software modules created in a variety of source code programming languages may be executed, and/or combined to form new programs. The programmer compiles source code (*e.g.*, written in a high-level programming language) into an intermediate language representation of the source code (*e.g.*, intermediate language or "IL" code) using a source compiler. In this case, the source compiler does not generate native (*e.g.*, machine executable) code, but instead provides only the IL representation. The runtime system then receives the IL code representation from the source compiler, and performs an IL to native code conversion, which may be accomplished using a just-in-time (JIT) compiler. This IL to native code conversion (*e.g.*, JIT compilation) may include combining software components (*e.g.*, modules, objects, *etc.*) from a variety of sources, which may have been originally coded in different high-level (*e.g.*, source code) languages.

B. Claims

Claims 1-11, 28-34, 41-43 and 47 stand rejected under 35 U.S.C. §102(e) as being anticipated by Alpern (US Patent 6,651,248). Applicant's representative respectfully submits that the claims as previously submitted are patentable over Alpern. However, in order to expedite prosecution, the claims have been amended to further clarify the claimed subject matter. Withdrawal of this rejection is respectfully requested for at least the following reasons. Alpern does not disclose compiling an association between a declaration and an implementation into its intermediate language representation and ***providing the intermediate language representation of the association between the declaration and the implementation to a target runtime to compile the intermediate language representation into native code***.

> A single prior art reference anticipates a patent claim if ***each*** and ***every*** limitation set forth in the patent claim is disclosed in the reference, either expressly or inherently. (*See Trintec Industries, Inc. v. Top-U.S.A. Corp.,* 295 F.3d 1292, 1295, 63 U.S.P.Q.2d 1597, 1599, 2 U.S.P.Q.2d 1051, 1052-53 (Fed. Cir. 2002) (citing to *Verdegaal Bros., Inc. v.*

> *Union Oil Co.*, 814 F.2d 628, 631 (Fed. Cir. 1987))) (emphasis added). Moreover, "[t]he ***identical*** invention must be shown in as ***complete*** detail as is contained in the patent claim." (*Richardson v. Suzuki Motor Co.,* 868 F.2d 1226, 1236, 9 U.S.P.Q.2d 1913, 1920 (Fed. Cir. 1989) (emphasis added) (citing *Jamesbury Corp. v. Litton Industrial Products, Inc.*, 756 F.2d 1556, 1560, 225 U.S.P.Q. 253, 257 (Fed. Cir. 1985); and *Connell v. Sears, Roebuck & Co.*, 722 F.2d 1542, 1548, 220 U.S.P.Q. 193, 198 (Fed. Cir. 1983))).

Alpern discloses an interface method table (IMT) comprising a table of entries that are used to support invocation of interface methods. (*See e.g.,* Alpern at Abstract). The entries each correspond to a set S of interface methods that are implemented by objects of a given class. (*See e.g.,* Alpern at Abstract). An IMT entry stores the following data: a) if the set S is empty, the IMT entry is a null/empty value; b) if the set S includes only a single interface method, the IMT entry is a pointer to the implementation of the single interface method in the set S; and c) if the set S includes two or more interface methods, the IMT entry is a pointer to a conflict resolution routine. (*See* Alpern at col. 3, ll. 20-31). The processing of a method invocation statement involves either 1) loading a pointer to the implementation of the interface method from an entry of the IMT and passing control to this implementation; or 2) loading a pointer to a conflict resolution routine pointed by an IMT entry and passing control to this conflict resolution routine. (*See* Alpern at col. 3, ll. 33-39). The conflict resolution routine pointed to by the IMT entry identifies *at execution time* the location of the particular interface method of the two or more interface methods that are associated with the given IMT entry. (*See* Alpern at Abstract and col. 3, ll. 39-45). Thus, the system of Alpern performs conflict resolution *at runtime*.

The claimed subject matter generally relates to methods and systems for implementing intermediate language representations of source code. (*See* Application at p. 3, l. 29 – p. 4, l. 16). A source complier compiles source code (*e.g.*, written in a high-level programming language) into an intermediate language representation of the source code (*e.g.*, intermediate language or "IL" code). (*See* Application at p. 2, ll. 14 – 16). The runtime system then receives the IL code representation from the source compiler, and performs an IL to native code conversion, for instance, using a just-in-time (JIT) compiler. (*See* Application at p. 2, ll. 18 – 19). Using override associations, the source compiler resolves potential ambiguities in

associating a declaration with an implementation. (*See* Application at p. 4, ll. 13 – 16; Fig. 11, elements 418, 450). Thus, at runtime, the expressed associations can be interpreted without encountering ambiguous associations. (*See* Application at p. 4, ll. 8 – 12).

All of the subject claims recite the limitations compiling an association between a declaration and an implementation into its intermediate language representation and ***providing the intermediate language representation of the association between the declaration and the implementation to a target runtime to compile the intermediate language representation into native code*** or similar limitations. Alpern does not disclose this novel feature of the subject claims. The Examiner contends that Alpern discloses providing an intermediate language representation of an association between a declaration and an implementation to a target runtime. (*See* Office Action dated June 5, 2006 at p. 5). Applicant's representative respectfully disagrees. As explained above, the system of Alpern performs conflict resolution at runtime.

The Examiner also contends that "the claims are not recited in a way to indicate that the intermediate language representation in the context of 'providing the intermediate language representation' contains the expressed associations, as the associations are expressed 'as an intermediate language representation.'" (*See* Office Action at pp. 11-12). Applicant's representative respectfully disagrees. The claims prior to the amendments submitted herewith clearly state that an association between a declaration and an implementation is expressed as an intermediate language representation and that this intermediate language representation is provided to the target runtime as do the amended claims.

In view of the foregoing, it is readily apparent that Alpern does not disclose each and every element of the subject claims. Accordingly, applicant's representative respectfully requests that the rejection be withdrawn.

CONCLUSION

The present application is believed to be in condition for allowance in view of the above comments and amendments. A prompt action to such end is earnestly solicited. In the event any fees are due in connection with this document, the Commissioner is authorized to charge those fees to Deposit Account No. 50-1063 [MSFTP212US]. Should the Examiner believe a telephone interview would be helpful to expedite favorable prosecution, the Examiner is invited to contact applicants' undersigned representative at the telephone number below.

Respectfully submitted,

AMIN, TUROCY & CALVIN, LLP

/Cheryl L. Young/
Cheryl L. Young
Reg. No. 43,298

AMIN, TUROCY & CALVIN, LLP
24TH Floor, National City Center
1900 E. 9TH Street
Cleveland, Ohio 44114
Telephone (216) 696-8730
Facsimile (216) 696-8731